VERSITA
ADVANCES IN MANUFACTURING SCIENCE AND TECHNOLOGY Vol. 37, No. 4, 2013
DOI: 10.2478/amst-2013-0031

THE PRIORITISATION OF PRODUCTION ORDERS UNDER THE BEE COLONY ALGORITHM

Andrzej Jardzioch, Katarzyna Bulwan

S u m m a r y

The paper presents the problem of determining the prioritisation of production orders. The proposed criterion function allows a comprehensive evaluation of various ways of prioritising taking into account both the income derived from the execution of production orders and the penalty for any delays which may occur. The criterion function was implemented in an algorithm based on the operation of a colony of bees. The experiments which have been carried out make it possible to evaluate the solutions obtained through the provided algorithm and compare them with the solutions obtained through the typical heuristic rules. The results show that the prioritisation obtained through the algorithm is characterized by the highest qualities of the criterion function and is definitely superior to that obtained through the simple heuristic rules.

Keywords: job shop scheduling, artificial bee colony algorithm, manufacturing systems

Szeregowanie zleceń produkcyjnych z zastosowaniem algorytmu pszczelego

S t r e s z c z e n i e

W pracy przedstawiono zagadnienia ustalania kolejności wprowadzania zleceń do produkcji. Zaproponowano zastosowanie kompleksowej funkcji kryterialnej do oceny różnorodnych uszeregowań. Funkcja ta uwzględnia zarówno przychód uzyskany z realizacji zleceń produkcyjnych, jak i ewentualne kary za opóźnienia w ich wykonaniu. Opracowano algorytm oparty na działaniu roju pszczół, w którym zaimplantowano proponowaną funkcję kryterialną. Wykonane eksperymenty pozwoliły na ocenę uszeregowań uzyskiwanych z użyciem algorytmu pszczelego oraz ich porównanie z rozwiązaniami dla typowych reguł heurystycznych. Analiza otrzymanych wyników pozwoliła na stwierdzenie, że uszere-gowania uzyskiwane z zastosowaniem opracowanego algorytmu cechowały się największymi wartościami funkcji kryterialnej. Zdecydowanie przewyższały uszeregowania uzyskiwane z wykorzysta-niem prostych reguł heurystycznych.

Słowa kluczowe: szeregowanie zleceń produkcyjnych, algorytm pszczeli, system wytwarzania

1. Introduction

Small and medium-sized industrial enterprises are very often forced to operate as sub-suppliers pursuing unit and small-series production. Such production is characterized by a large number of small quantity production

Address: Andrzej JARDZIOCH, D.Sc. Eng., Katarzyna BULWAN, MSc Eng, West Pomeranian University of Technology, Faculty of Mechanikal Engineering and Mechatronics, Piastów 19, 70-310 Szczecin, Poland, e-mail: Andrzej.Jardzioch@zut.edu.pl; Katarzyna.Bulwan@zut.edu.pl

orders with clearly defined deadlines and the possible penalties for not meeting them [1]. In such market conditions it is necessary to properly plan the sequencing of particular production orders so that their execution could bring maximum economic benefits. The primary task is to satisfy customers' expectations by meeting the deadlines of the orders and at the same time keeping the production costs as low as possible [2-4].

The tasks of planning and scheduling production processes are carried out in companies by specialized planning teams, more and more often using complex computer systems, ERP – Enterprise Resource Planning and MES – Manufacturing Execution Systems [5, 6]. Despite enormous capabilities of modern computer systems used in the planning processes, the problem of the quick prioritisation of production orders is still not solved. The commonly used ERP systems and MES systems, which cooperate with them, help to manage storage, collect production information, calculate in detail and monitor production costs. Relatively poorly developed are the modules responsible for the effective prioritisation of production orders and the proper attribution of particular orders to machines.

The article presents the idea of using the bee colony algorithm to solve the problem of prioritising orders in the production process of cutting elements of any shape out of metal sheets. The criterion for the sequence of production orders is the objective function taking into account the net profit gained from the execution of the order and the possible penalty in case of not meeting the deadline. Two problems of sequencing orders have been defined. The first problem is to determine the sequence of orders in the fixed set of production orders (static prioritising of production orders). In the second case the set of production orders is changeable (dynamic problem of prioritising orders). Under such approach it is assumed that during the introduction of orders to production there are new orders that are attached to the set of production orders.

2. Account of the problem

The problems connected with the prioritisation of production orders is an extremely complex issue which is discussed in many scientific publications. Because of the classification of this problem into NP-hard problems, the proposed solutions are based on meta-heuristic algorithms such as the Tabu Search procedure, fuzzy logic, evolutionary algorithms, simulated annealing and other artificial intelligence methods [7, 8]. Scientific work connected with the problem of the prioritisation of production orders focuses on two issues. The first one consists in developing specialized accurate methods leading to optimal solutions. As an example we might mention here branch and bound methods, an integer linear programming, dynamic programming, Johnson's algorithm [9, 10]. An important advantage of this group of methods is the possibility of finding the optimal solution, but they also have very big disadvantages. They can only be used for solving specific problems and consequently their application is limited. Therefore, to solve the problems of the prioritisation of orders, approximate methods, also known as heuristic and meta-heuristic techniques, are increasingly being used [10-12]. The typical heuristic algorithms are based on simple rules of priorities (the so-called structural algorithms). The following rules can be listed:

- Shortest Processing Time – SPT, Longest Processing Time – LPT,
- Most Work Remaining – MWR, Least Work Remaining – LWR ,
- Most Operations Remaning – MOR, Least Operations Remaning – LOR,
- Earliest Due Date – EDD,
- First Come First Served FCFS, Last Come First Served – LCFC.

A very innovative group of methods used for solving the problems of prioritising production orders are meta-heuristic methods based on the adaptation of the mechanisms operating in nature. They include: evolutionary algorithm, ant algorithm, immune algorithm, simulated annealing, fuzzy logic and bee colony algorithm [9, 11, 13-15].

The article presents the problem of the prioritisation of production orders in a flexible production system of cutting elements out of metal sheets. (Fig. 1).The most important components of the analysed system is jet cutter M_1 and automated shelved metal storehouse M_{in}. Production orders are submitted by customers via the Internet and transmitted to the master planning subsystem where they are automatically prioritised and successively transferred to the execution. After the treatment, the executed orders are stored in output storehouse M_{out}.

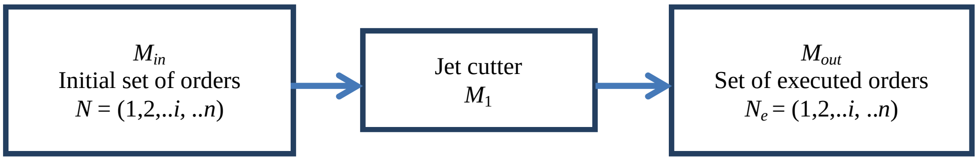

Fig. 1. Flexible production system

Two approaches to the problem of the prioritisation of production orders have been defined. The first one assumes that the set of orders accepted for production is constant. It means that once sorted orders are transmitted to the production process where they are executed in the accepted order. Only after the execution of the whole set of orders, the next set of lichen orders is introduced. This approach is called statistical prioritising of production orders. The second approach is characterized by a continuous introduction of new orders to the set of orders provided for production. In this case, after sorting the set of orders for production, only the first order in the set is introduced. The time in which the introduction of the next order for production is decided, the set of production orders is sorted again, including the orders which have since been accepted. This approach is called dynamic prioritising of production orders. For the prioritisation of production orders the bee colony algorithm has been implemented. The results obtained through the algorithm have been compared with the results obtained through the EDD and FCFS rules.

Due to the fact that the system works in the conditions of unit and small-series production, there is a large variety of production orders. Production order z_i is described by defining the shape of the cut element, the thickness of the metal sheet from which the element is cut, the number of pieces of cut-out elements, the required time limit for completion $d_w(z_i)$, the income derived from the execution of the order $r(z_i)$ and the indicator of the penalty $w(z_i)$ in case of not meeting the deadline. The penalty for the delay is charged when the actual due date of the execution of the order denoted as $C(z_i)$ is longer than the planned date of the execution of the order $dw(z_i)$ (that is, when $C(z_i) – dw(z_i) > 0$). When actual due date $C(z_i)$ is shorter than planned date $d_w(z_i)$, the delay does not occur and the penalty will not be charged. It is assumed that the penalty for the delay is proportional to the time of the delay and the penalty indicator.

The objective of the prioritisation of production orders is to determine such a schedule of their execution that it could make the derived income as big as possible, that is to make the function describing the sum of the revenue generated from the execution of all the orders minus the possible penalty for the delay reach the maximum. The objective function is shown in Formula (1)

$$fit(r_i, d_i, w_i, t_i) = \sum_{i=1}^{N} r_i - w_i * \max\{0, C_i - d_i\} \quad (1)$$

where N – number of orders, r_i – revenue from order; d_i – required due date of order i; w_i – gravity of penalty of order i; C_i – actual due date of order i.

3. The principle of the bee colony algorithm

The Artificial Bee Colony algorithm ABC is a meta-heuristic algorithm applied to the optimization of numerical problems by Karabog in [16]. It belongs to the group of colony algorithms, which represent a broad class of optimization algorithms drawing their inspiration from nature, and more specifically, from the behaviour of honey bees during the search for food.

In the wild, the organization of activities of bees wanting to get the maximum amount of food consists in sending in all directions spotter bees which fly up to 10 km in search of meadows rich in flowers suitable for use. After exploring the area, the bees go back to the hive and in a specific way by the so-called waggle dance report on the situation to other bees, giving three basic items of information from their reconnaissance: the direction from which the bees came (the angle between the sun and the source of their food), the distance to the meadows and the abundance of food resources. On the basis of the data describing the quality of the meadows (distance and abundance), the bees in the hive decide to set out to the appropriate meadow. Returning with pollen from the meadow, worker bees also perform a dance in the hive to update on the exploited meadow [15-18].

The ABC algorithm consists in using the behaviour of bee colonies in order to find the optimal prioritisation of tasks. The operation of the algorithm is based on the local searching connected with random mechanisms moving the area of the search into new areas. The first step in the operation of the algorithm is to generate new initial solutions which in the subsequent stages are analysed and then updated. The solutions and information about the value of the objective function are stored by bees modelled within the algorithm. The updating takes place when the new adjacent solution obtained through adjacent exchange aEX is better than the initial solution. The operation of the adjacent exchange is to choose at random two adjacent positions k and $k + 1$ in the solution vector ω and exchange them. The result of the operation is the generated new solution vector ω. The comparison of the solutions is done through the quality function fit(r_i, d_i, w_i, t_i) defined according to formula 1.

The developed algorithm based on the idea of the bee colony algorithm consists of many steps, the most important being the step of randomly generating input solutions and selecting the global initial solution, the step of generating the adjacent solution through the adjacent exchange and the step of drawing the active bee through the roulette function and collecting the solution stored by the bee. Using the draw in the form of the roulette function allows the introduction of the mechanism in which the probability of selecting the bee with the information about a better solution is bigger than the probability of selecting the bee with a worse solution.

The complete pseudo-code of the ABC algorithm is presented in a simple form in Fig. 2. The basic parameters of the ABC algorithm are: number of cycles (C), set of bees (P), set of active bees (X), set of inactive bees (Y), number of visits (L). The parameter describing the set of bees (P) makes it possible to define the initial solutions generated at random. The set of active bees (X) is used to evaluate solutions in the way specified for active bees (point 3.1.1. of the pseudo-code). The set of inactive bees (Y) is used to evaluate solutions in the way described in the point 3.1.2. of the pseudo-code. The parameter of the number of visits (L) defines the number of evaluations after which the initial solution, which did not reach higher values of the objective function, will be replaced by another output solution drawn at random (points 3.1.1.3. and 3.1.2.3. of the pseudo-code).

1. Load the parameters of the colony (*C* – number of cycles, *P* – number of bees, *X* – number of active bees, *Y* – number of inactive bees, *L* - max. number of visits).
2. Assign each of bees P a random solution and after calculating quality save the best of them as the best global solution.
3. For each: *cycle = 0, 1, 2, … , C-1*
   3.1. For each: *bee = 0, 1, 2, … , P-1*
   3.1.1. If the bee is active (belongs to *X*)
      3.1.1.1 Define the adjacent solution from the solution currently stored by the bee through the adjacent exchange.
      3.1.1.2. Exchange the old solution for a new one if it is better and set the counter of visits *counter = 0*. Otherwise enlarge the counter of visits by 1, *counter = counter + 1*.
      3.1.1.3. If *counter = L* draw a new solution and replace the previous one.
      3.1.1.4. Check whether the new solution is better than the best one . If so, save it as the best global one.
   3.1.2. If the bee is inactive (belongs to *Y*)
      3.1.2.1. Draw the active bee using the roulette wheel selection and pick the solution stored by it.
      3.1.2.2. Define the adjacent solution from the solution currently stored by the bee through the adjacent exchange.
      3.1.2.3. Exchange the old solution for a new one (if it is better) and set the counter of visits *counter = 0*. Otherwise enlarge the counter of visits by 1, *counter = counter +1*.
      3.1.2.4. If *counter = L,* draw a new solution and exchange it for the previous one.
      3.1.2.5. Check whether the new solution is better than the best one. If so, save it as the best global one.
4. Display the best solution.

Fig. 2. Pseudo-code showing the principle of the implemented algorithm ABC

In each subsequent iteration of the algorithm, it is checked whether the found solution is better than the solution currently stored as the global solution. When the found solution is better, the global solution is updated. In the developed algorithm there are reductions in the adjacent search. Each bee, searching for a new solution on the basis of the currently stored solution, continues the search until a better solution is found or when the defined search limit is exceeded.

4. Research and the results of the computational experiment

The developed algorithm based on meta-heuristics simulating the behaviour of a bee colony became the basis for the implementation of a computer system dedicated to the issues of the prioritisation of production orders. The program was written in C# using the Microsoft Visual Studio 2012. With this tool it was decided to carry out research aimed at comparing the quality of the prioritisation of production orders through the ABC algorithm and typical heuristic rules (EDD and FCFS). To evaluate the quality of the obtained solutions, they were compared with the optimal solutions found through the complete reviewing method. The evaluation was conducted on the basis of the criterion function shown in Equation 1.

During the first stage of the research a static approach to the prioritisation of production orders was adopted. Sets of orders of different sizes were assumed: N = 7, 8, 10, 15, 20. Each order was assigned random values of time required to execute order d_i in the range of (7:30), income from the execution of order r_1 in the range of (25:80), required time limit within which the order should be executed C_j in the range of (3:14), indicator of penalty for a delay in completing order w_i in the range of (0,1:0,9). With the developed bee colony algorithm the following parameters were used: number of cycles C = 3460, number of bees P = 50, number of active bees X = 25, number of inactive bees Y = 25, maximum number of visits L = 100. In order to average the results, each time calculations were carried out for four different sets of data. The results obtained are summarized in Table 1.

The analysis of the results shown in Table 1. leads to the conclusion that the prioritisation generated through the ABC algorithm reaches by far the highest value of the criterion function. It is worth noting that the orders prioritised through the ABC algorithm are at the same time the optimal solutions, which was checked by conducting the complete reviewing of all the solutions (for the sets of orders N = 7,8,10). The solutions obtained through the rules of EDD and FCFS differ significantly from the optimal solutions.

In the second stage of the research, the efficiency of the chosen algorithms through dynamic prioritising of production orders was compared. Under the accepted scenario, five moments were defined when it was necessary to provide follow-up orders for production (T_1, T_2, T_3, T_4, T_5). At time T_1 there were 15 orders in the production system. Using the particular algorithms, three different ways of prioritising were obtained. The orders that were in the first place of each prioritisation, were directed to production. In the course of the execution of the first order directed to production, two more orders were received by the system (orders 15 and 16). This made it necessary at time T_2 to generate a new way of prioritising so that another production order could be designated. An analogous procedure took place at time T_3 (additional orders 18 and 19), at time T_4 (orders 20 and 21) and at time T_5 (orders 22 and 23). The scenario describing the process of receiving subsequent orders with their parameters is shown in Table 2.

Table 1. Comparison of the effects of prioritising

Number of production orders in set N	**Objective function value for different prioritising methods**				**Objective function value for complete reviewing**
	Number of experiment	**ABC**	**EDD**	**FCFS**	
7	1	375,6	372,1	370,1	375,6
	2	287,9	264,3	255,4	287,9
	3	339,1	322,4	300,8	339,1
	4	470,7	464,6	443,0	470,7
8	1	428,5	431,5	402,6	428,5
	2	352,8	314,4	299,6	352,8
	3	359,9	323,5	306,5	359,9
	4	405,4	391,7	373,2	405,4
10	1	498,2	477,5	468,9	498,2
	2	317,8	263,1	272,3	317,8
	3	435,4	374,2	371,0	435,4
	4	551,3	494,0	487,4	551,3
15	1	549,8	414,6	405,9	b.d.
	2	660,6	525,5	480,4	b.d.
	3	540,1	503,5	441,7	b.d.
	4	557,4	362,9	496,8	b.d.
20	1	702,5	474,7	464,8	b.d.
	2	709,7	548,4	612,0	b.d.
	3	761,4	582,4	541,3	b.d.
	4	651,7	540,1	651,4	b.d.

The choice of the order directed to production depended on the algorithm used. For example, at time T_2 , using the ABC algorithm allowed the choice of order Z_9, using the EDD algorithm made it possible to choose order Z_{13} and the FCFS algorithm – order Z_2. To finish the research experiment it was assumed that the way of prioritising developed at time T_5 was introduced to production in an unchanged form. Table 3 presents the prioritisation of production orders at moments T_1, T_2, T_3, T_4 and T_5, obtained through the three analysed algorithms. The use of the ABC algorithm allowed prioritisation (2,16,9,3), the EDD algorithm – prioritisation (2,13,4,12,16) and the FCFS algorithm – prioritisation (1,2,3,4,5). The execution of orders through those ways of prioritising obviously resulted in obtaining a variety of aggregated income. Also this time, the biggest income was obtained through the ABC algorithm used for prioritising.

Table 2. Scenario of accepting subsequent production orders

Number of order	Profit	Penalty indicator	Processing time	Due date
Time T_1, initial set of production orders				
1	27	0.2	8	18
2	74	0.6	5	7
3	61	0.4	3	18
4	66	0.2	4	11
5	77	0.3	12	14
6	59	0.4	9	16
9	64	0.1	6	26
8	54	0.3	9	26
9	62	0.7	7	16
10	29	0.1	4	29
11	31	0.7	6	17
12	72	0.1	4	12
13	67	0.2	4	10
14	36	0.5	3	19
15	58	0.2	11	26
Time T_2, two new orders accepted				
16	37	0.6	9	12
17	60	0.3	13	15
Time T_3, two new orders accepted				
18	29	0.1	10	20
19	70	0.7	13	28
Time T_4 two new orders accepted				
20	40	0.3	9	13
21	51	0.2	10	21
Time T_5, two new orders accepted				
22	51	0,3	4	17
23	40	0,6	10	13

5. Conclusion

The paper presents two approaches to the prioritisation of production orders. The first approach assumes a static nature of the process of prioritising characterized by a fixed set of production orders subjected to prioritisation. The second approach takes into account the dynamics of the actual process of prioritising, in which the set of production orders is continuously replenished with subsequent orders. In both cases the problem of determining the quality of the developed prioritising methods is of great importance. The paper defines the criterion function allowing a comprehensive evaluation of different ways of prioritising production orders, taking into account both the revenue gained from their execution and a possible penalty for the delay.

Table 3. Comparison of prioritising methods used at moments T_1, T_2, T_3, T_4, T_5

No.	**Type of prioritising algorithm used**														
	ABC					**EDD**					**FCFS**				
	T1	T2	T3	T4	T5	T1	T2	T3	T4	T5	T1	T2	T3	T4	T5
1	**2**	**16**	**9**	**3**	**23**	**2**	**13**	**4**	**12**	**16**	**1**	**2**	**3**	**4**	**5**
2	9	9	11	13	**11**	13	4	12	16	**20**	2	3	4	5	**6**
3	11	14	3	4	**14**	4	12	16	20	**23**	3	4	5	6	**7**
4	14	3	14	11	**22**	12	16	5	5	**5**	4	5	6	7	**8**
5	3	11	19	14	**19**	5	5	17	17	**17**	5	6	7	8	**9**
6	13	13	13	19	**4**	6	17	6	6	**6**	6	7	8	9	**10**
7	4	4	4	6	**13**	9	6	9	9	**7**	7	8	9	10	**11**
8	6	6	6	8	**6**	11	9	11	11	**11**	8	9	10	11	**12**
9	8	8	8	20	**20**	1	11	1	1	**22**	9	10	11	12	**13**
10	5	1	5	5	**8**	3	1	3	3	**1**	10	11	12	13	**14**
11	1	10	1	10	**12**	14	3	14	14	**3**	11	12	13	14	**15**
12	10	12	10	12	**5**	7	14	18	18	**14**	12	13	14	15	**16**
13	12	5	12	1	**1**	8	7	7	21	**18**	13	14	15	16	**17**
14	15	17	17	17	**10**	15	8	8	7	**21**	14	15	16	17	**18**
15	7	15	15	21	**17**	10	15	15	8	**9**	15	16	17	18	**19**
16		7	7	15	**21**		10	19	15	**8**		17	18	19	**20**
17			18	7	**15**			10	19	**15**			19	20	21
18				18	**7**				10	**19**				21	**22**
19					**18**					**20**					**23**
Aggregated income					**3642,6**	**3592,8**					**3567,6**				

To prioritise production orders the meta-heuristic algorithm based on the operation of a bee colony was developed, into which original extensions that define the solution by means of the adjacent exchange were introduced. With the use of the developed algorithm a computer program for carrying out simulation experiments was built. The objective of the experiments was to determine the quality of the prioritisation generated through the developed bee colony algorithm (ABC). On the basis of the conducted experiments it can be stated that the solutions obtained through the ABC algorithm coincide with the optimal solutions obtained as a result of the complete reviewing.

References

[1] C.-H. LAN: The design of multiple production lines under deadline constraint. *Int. Journal of Production Economics*, **106**(2007), 191-203

[2] Y.-H. HSIEH, S.-T. YUAN, R.-L. KUO: A PSO-based intelligent service dispatching mechanism for customer expectation management. *Expert Systems with Applications*, **38**(2011), 12128-12141.
[3] H. GÜNTHER, P. BEEK (Eds.): Advanced planning and scheduling solutions in process industry. Springer-Verlag, Berlin, Heidelberg, 2003.
[4] J. SUN, D. XUE: A dynamic reactive scheduling mechanism for responding to changes of production orders and manufacturing resources. *Computers in Industry*, **46**(2001), 189-207.
[5] V. BOTTA-GENOULAZ P. MILLET: A classification for better use of ERP systems. *Computers in Industry*, **56**(2005), 573-587.
[6] J.M. SIMÃO, P.C. STADZISZA, G. MOREL: Manufacturing execution systems for customized production. *Journal of Materials Processing Technology*, **179**(2006), 268-275.
[7] S.C.L. KOHA, S.M. SAADB: The use of intelligent feedback for work order release in an uncertain manufacturing system. *Robotics and Computer-Integrated Manufacturing*, **20**(2004)6, 517-527.
[8] O. LÓPEZ-ORTEGA, I. VILLAR-MEDINA: A multi-agent system to construct production orders by employing an expert system and a neural network. *Expert Systems with Applications*, **36**(2009)2, 2937-2946.
[9] A. ALLAHVERDI, N.D. JATINDER GUPTA, T. ALDOWAISAN: A review of scheduling research involving setup considerations. *Omega, Int. J. Mgmt Sci.*, **27**(1999), 219-239.
[10] A. ALLAHVERDI, C.T. NG, T.C.E. CHENG, M. Y. KOVALYOV: A survey of scheduling problems with setup times or costs. *European Journal of Operational Research*, **187**(2008), 3985-1032.
[11] J. BŁAŻEWICZ, W. DOMSCHKE, E. PESCH: The job shop scheduling problem: Conventional and new solution techniques. *European Journal of Operational Research*, **93**(1996), 1-33.
[12] J. JÓZEFOWSKA, A. ZIMNIAK, Optimization tool for short-term production planning and scheduling. *Int. Journal of Production Economics*, **112**(2008)1, 109-120.
[13] A. R. YILDIZ: A new hybrid artificial bee colony algorithm for robust optimal design and manufacturing. *Applied Soft Computing*, **13**(2013), 2906-2912.
[14] P.P. ANGELOV: Evolving rule-based models. Tool for decision of flexible adaptive systems. Physica-Verlag, Wurzburg 2002.
[15] B.-C. CHEN, J.Y. LIN, T.-P. HONG: Learning discriminate functions with fuzzy attributes for classification using genetic programming. *Expert Systems Appl*., **23**(2002), 31-37.
[16] D. KARABOGA: An idea based on honey bee swarm for numerical optimization. Technical Report-TR06, Erciyes University, 2005.
[17] X. WANG, X. XIE, T.C.E. CHENG: A modified artificial bee colony algorithm for order acceptance in two-machine flow shops. *Int. Journal Production Economics*, **141**(2013), 14-23.
[18] R. ZHANG: An artificial bee colony algorithm based on problem data properties for scheduling job shops. *Procedia Engineering*, **23**(2011), 131-136.

Received in June 2013